CST1
COMPUTER SCIENCE TRIPOS Part IB

Tuesday 4 June 2024 13:30 to 16:30

COMPUTER SCIENCE Paper 5

*Answer **five** questions.*

*Submit the answers in five **separate** bundles, each with its own cover sheet. On each cover sheet, write the numbers of **all** attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
Approved calculator permitted

1 Computer Networking

http v1.2
TCP
IPv4
IEEE 802.11 (WiFi)
2.4 GHz wireless band

Consider this communications-stack diagram. Each layer of the stack can provide multiplexing support to the layer above, mapping a single channel into multiple independent communications channels.

For each of the following examples of the diagram's communications-stack layers, discuss how multiplexing is put into practice, including an example of how relevant data is identified and how such data is mapped to and from the layer above.

(*a*) A 2.4 GHz physical layer. [4 marks]

(*b*) A data-link layer. [4 marks]

(*c*) An IPv4 network layer. [4 marks]

(*d*) A TCP transport layer. [4 marks]

(*e*) A web server using HTTP v1.2. [4 marks]

2 Computer Networking

(*a*) Physical line coding, for fiber or copper, discusses the design of systems that transform digital communications into an analog channel; the analog channel is often referred to as the physical layer.

(*i*) Compare and contrast the two physical line coding systems: a pure scrambling system, and a pure block codec. [6 marks]

(*ii*) For each of scrambler and block code, outline the disadvantages in their sole use. Then indicate how a hybrid schema can solve such drawbacks. [4 marks]

(*b*) Wireless networking faces several challenges due to the physics of radio communications. Explain, with the use of illustration, two distinct collision-detection issues unique to wireless networks, and then describe a protocol-centered approach that will overcome each issue. [10 marks]

3 Computer Networking

(*a*) Define the terms latency and capacity as applied to communication channels [2 marks]

(*b*) Is there a strict relation between the two? [1 mark]

(*c*) Imagine a packet-based network channel, with fixed bandwidth and fixed latency, providing service to a transport protocol.

(*i*) In this context, describe a basic system that provides reliable transmission of packet data. Include a description of the essential features of your approach. [8 marks]

(*ii*) The network channel will impact the effective throughput of your reliable transport protocol. Outline an example of why this occurs. [2 marks]

(*iii*) Describe a straightforward extension to your protocol that would improve the performance of your transport protocol without changing the channel provided by the underlying network. [4 marks]

(*iv*) In what circumstances might these improvements have only limited benefit? [3 marks]

4 Concurrent and Distributed Systems

(*a*) A concurrency library provides a `signal` primitive for semaphores and a `notify` primitive for condition variables. Explain what they have in common and how they are different. Are they blocking primitives? Can they have no effect? [4 marks]

(*b*) A transaction processing system uses non-strict isolation.

(*i*) Without using an example, define the notion of serialisability of transactions. [2 marks]

(*ii*) What is a conflict between two threads, and can conflicts be used to check for serialisability? [2 marks]

(*iii*) What are the advantages and disadvantages of non-strict isolation? [2 marks]

(*iv*) Can performance be enhanced when increment and decrement operations are considered as composite operations? Consider this in the context of conflict analysis techniques. [2 marks]

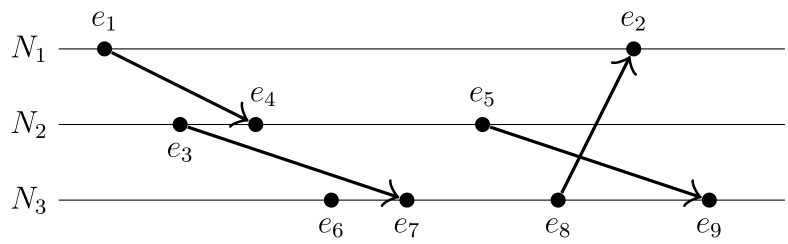
(*v*) Give an example, not using increments or decrements, where simplistic conflict analysis may report a problem that in reality is not a problem. [2 marks]

(*c*) Some code is replaced in an asynchronous, reliable message-passing language that uses `channel_id!value` and `?channel_id` to communicate between user-space threads. The whole system has now seized up. Sketch two possible code fragments that may have caused the seizure, one due to deadlock, and one not due to deadlock. [6 marks]

5 Concurrent and Distributed Systems

Distributed systems often use *logical timestamps* to capture the possible ordering between operations on different nodes.

(*a*) Three nodes $N_1 \dots N_3$ communicate using point-to-point messages:



(*i*) Give *Lamport timestamps* for each of the events $e_1 \dots e_9$, assuming clocks start from 0. [2 marks]

(*ii*) What is a necessary condition on Lamport timestamps for an event x to have *happened before* an event y? Is this a sufficient condition? Briefly justify your answer based on the events in the example. [2 marks]

(*iii*) Now give possible *vector timestamps* for each of the events $e_1 \dots e_9$. [2 marks]

(*iv*) What is a *necessary and sufficient* condition on vector timestamps for an event x to have happened before an event y? [2 marks]

(*b*) An alternate form of vector timestamps can be used to build *causal broadcast*. That is, if the broadcast of one message happened before the broadcast of another message, then all nodes must deliver those two messages in that order.

(*i*) Write pseudo-code to implement causal broadcast, assuming access to an underlying reliable broadcast protocol. You should describe the local state held at each node along with the algorithm used to broadcast a message, and to decide when a message can be delivered locally. [8 marks]

(*ii*) A colleague suggests that the space required by vector timestamps makes this algorithm inefficient. They suggest basing the algorithm on Lamport timestamps instead. Do you agree with this suggestion? [4 marks]

6 Introduction to Computer Architecture

The SystemVerilog code below is for a simple input/output processor that is focused on handling state changes on input and output pins.

```
typedef enum { CONST, JMP, BNZ, PUSH,
               SET, GET, INC, DEC } opcode;
typedef enum { RX, RY } regname;
typedef logic [31:0] word;
typedef logic [3:0]  immediate;
typedef logic [3:0]  pins;
typedef struct {opcode op;
                regname r;
                immediate i; } instruction;
module pio_proc
  (
   input  logic   clk,
   input  logic   rst,
   input  pins    inpins,
   output pins    outpins,
   output word    outw
   );
instruction imem[0:15] =
  '{'{op:CONST, r:RY, i:4'd0},  // 0
    '{op:GET,   r:RX, i:4'd3},  // 1
    '{op:DEC,   r:RX, i:4'd0},  // 2
    '{op:BNZ,   r:RX, i:4'd1},  // 3
    '{op:GET,   r:RX, i:4'd3},  // 4
    '{op:BNZ,   r:RX, i:4'd4},  // 5
    '{op:INC,   r:RY, i:4'd0},  // 6
    '{op:GET,   r:RX, i:4'd3},  // 7
    '{op:DEC,   r:RX, i:4'd0},  // 8
    '{op:BNZ,   r:RX, i:4'd6},  // 9
    '{op:INC,   r:RY, i:4'd0},  // 10
    '{op:GET,   r:RX, i:4'd3},  // 11
    '{op:GET,   r:RX, i:4'd3},  // 12
    '{op:BNZ,   r:RX, i:4'd10}, // 13
    '{op:PUSH,  r:RY, i:4'd0},  // 14
    '{op:JMP,   r:RX, i:4'd0}}; // 15

instruction ir;
immediate pc;
word rf [regname];
always_comb ir = imem[pc];
always_ff @(posedge clk or posedge rst)
  if(rst)
    pc <= 0;
  else begin
    case(ir.op)
      CONST: rf[ir.r] <= {28'b0, ir.i};
      JMP:   pc <= ir.i;
      BNZ:   pc <= (rf[ir.r]!=0) ? ir.i : pc+1;
      PUSH:  outw <= rf[ir.r];
      SET:   outpins[ir.i[1:0]] <= rf[ir.r][0];
      GET:   rf[ir.r] <= {31'b0,inpins[ir.i[1:0]]};
      INC:   rf[ir.r] <= rf[ir.r]+1;
      DEC:   rf[ir.r] <= rf[ir.r]-1;
    endcase // case (ir.op)
    if((ir.op!=JMP) && (ir.op!=BNZ))
      pc <= pc+4'd1;
  end
endmodule // pio_proc
```

(*a*) How many clock cycles does it take to fetch, decode and execute an instruction? Explain your answer. [2 marks]

(*b*) What are the eight instructions and what function do they each perform? Give a short English explanation of each. [6 marks]

(*c*) The `imem` memory is initialised with a program. Produce commented pseudo code that represents the program. [6 marks]

(*d*) In the SystemVerilog module, what registers are not reset and will they cause the processor to malfunction? [3 marks]

(*e*) What is the meaning of the `outw` output? Is the instruction timing of the program important? [3 marks]

7 Introduction to Computer Architecture

A microprocessor design team currently has a classical five-stage pipelined processor comprising: instruction fetch, decode + register fetch, execute, data memory access, write-back. They also included independent instruction and data level-1 caches (32 KiB each) and one level-2 cache (256 KiB).

(*a*) In the context of this five-stage pipeline, explain what the following hazards are and how they might arise:

(*i*) control hazards, [2 marks]

(*ii*) structural hazards. [2 marks]

(*b*) For this five-stage pipeline, what are forwarding paths and why they are needed? Include a sketch of the pipeline indicating where the forwarding paths are located. [3 marks]

(*c*) To boost performance, new 128 KiB level-1 instruction and data caches are proposed, but these need to be pipelined to maintain the clock frequency resulting in 2-cycle access latency. The level-2 cache is boosted to 1 MiB without increasing the level of pipelining. This takes the five-stage pipeline to seven-stages.

(*i*) Sketch the new pipeline. Why might new forwarding paths be required? [3 marks]

(*ii*) What is a load-to-use penalty for the five-stage and seven-stage pipelines? [2 marks]

(*iii*) What is a branch penalty for the five-stage and seven-stage pipelines? [2 marks]

(*iv*) What is the cache access latency in cycles for each design given the following table of parameters? [4 marks]

	pipeline size	
	5-stage	7-stage
Level-1 data-cache miss rate	5%	3%
Level-2 cache miss rate	2%	1.5%
Level-1 data-cache access latency in cycles	1	2
Level-2 cache access latency in cycles	9	9
DRAM access latency in cycles	100	100

(*d*) What is tandem verification and how could the technique be used to test the pipelined processor designs? [2 marks]

8 Introduction to Computer Architecture

(*a*) Why do multicore systems-on-chip use hierarchical caches rather than one large shared cache? [3 marks]

(*b*) Why are page tables hierarchical rather than flat? Illustrate your answer by considering the memory required to store the page table for:

- an application with a contiguous 6 MiB memory footprint starting at virtual address zero,
- running on a 32-bit processor, with a 32-bit physical address space and 4 KiB pages. [5 marks]

(*c*) What is a TLB and why do we need one per core? [3 marks]

(*d*) What is the difference between inclusive, exclusive and non-inclusive non-exclusive (NINE) cache inclusion policies? [3 marks]

(*e*) In multicore system with private level-1 and level-2 caches and one shared level-3 cache, why might the level-2 cache be inclusive but the level-3 cache be NINE? [3 marks]

(*f*) What is the difference between cache coherence and consistency? Give examples of two consistency models. [3 marks]

END OF PAPER